PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-035882
(43)Date of publication of application : 02.02.2000

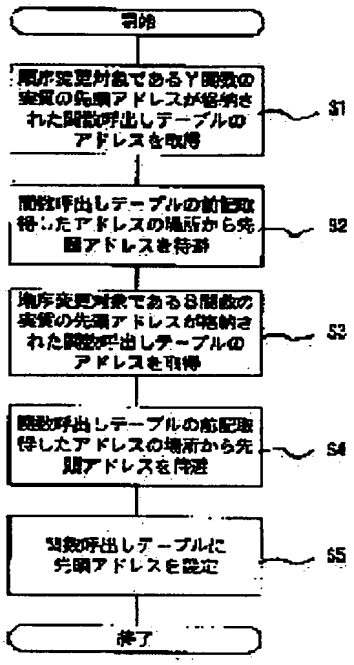
(51)Int.Cl. G06F 9/06

(21)Application number : 10-204909
(22)Date of filing : 21.07.1998

(71)Applicant : OKI SOFTWARE OKAYAMA:KK
OKI ELECTRIC IND CO LTD
(72)Inventor : SHIRANE TOSHIO
SHIRANE KYOKO
SHIGEHIRO MASAO

(54) METHOD FOR DYNAMIC CHANGE OF FUNCTION CALLING SEQUENCE

(57)Abstract:
PROBLEM TO BE SOLVED: To dynamically change a calling sequence of the function of a different product by setting the head address of the execution part of the function being the object of an order change to a prescribed place in a function call table.
SOLUTION: The address of a function call table where the head address of the substantial execution part of Y function being one of the objects of an order change is stored is obtained. The stored head address of the substantial execution part of Y function is acquired and the head address is saved from the function call table (S1 and 2). The address of the function call table where the head address of the substantial execution part of B function being the other of the objects is stored is obtained. The stored head address of the substantial execution part of B function is obtained and the head address is saved from the function call table (S3 and 4). The head address of the substantial execution part of B function is set in the table address and the head address of the substantial execution part of Y function is set in the table address (S5). Then, the function call table is rewritten.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2000－35882
（P2000－35882A）
(43)公開日　平成12年２月２日(2000.2.2)

(51)Int.Cl.'	識別記号	ＦＩ		テーマコード'（参考）
Ｇ０６Ｆ　9/06	５４０	Ｇ０６Ｆ　9/06	５４０Ｅ	５Ｂ０７６

審査請求　未請求　請求項の数６　ＯＬ　（全　９　頁）

(21)出願番号　特願平10－204909

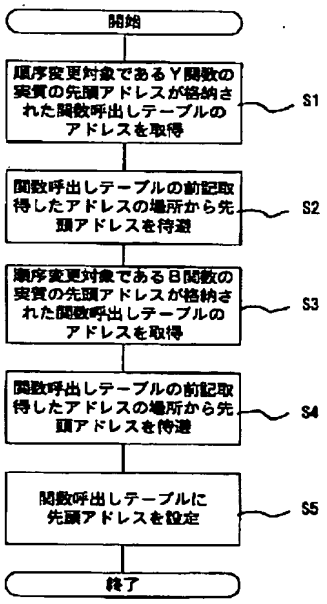
(22)出願日　平成10年７月21日(1998.7.21)

(71)出願人　593205679
株式会社沖ソフトウェア岡山
岡山県岡山市桑田町18番28号
(71)出願人　000000295
沖電気工業株式会社
東京都港区虎ノ門１丁目７番12号
(72)発明者　白根　寿雄
岡山県岡山市桑田町18番28号　株式会社沖ソフトウェア岡山内
(74)代理人　100061273
弁理士　佐々木　宗治　（外３名）

最終頁に続く

(54)【発明の名称】　関数呼び出しの手順を動的に変更する方法

(57)【要約】
【課題】　第１のプログラム内の関数を呼び出す手順を所望の手順に変更することが可能な関数呼出しの手順を動的に変更する方法を提供する。
【解決手段】　自身のプログラムの実行中に、関数呼び出しテーブル書き換え処理に基づいて、関数呼び出しテーブルに格納されている先頭アドレスのうち、順序変更の対象である関数Ｙ及び関数Ｂの実質の実行部の先頭アドレスを、関数呼び出しテーブル内の所定の場所に設定して、関数呼び出しの手順を動的に変更するものである。



本発明の一実施の形態の処理の流れを示すフローチャート

1

【特許請求の範囲】

【請求項１】　実行形式を変更できない第１のプログラムに含まれる関数のうち、順序変更対象の関数が前記第１のプログラムにおいて呼び出される順序が予め判明している場合に、前記第１又は該第１のプログラムを使用した第２のプログラム内の関数の呼び出しを追加することによって、第１のプログラム内の関数を呼び出す順序を前記第２のプログラムの実行中に変更する方法であって、

第２のプログラムにおいて呼び出される関数への呼び出し先が格納される関数呼び出しテーブルを備え、前記追加される関数は再帰的な関数であり、そして、前記第２のプログラムは、前記関数呼び出しテーブルのうち、順序変更対象の関数への呼び出し先が格納されている場所に、予め判明している前記呼び出し順序に基づいて、前記場所に格納された呼び出し先を入れ替えて格納することによって関数呼び出しテーブルを書き換える関数呼び出しテーブル書き換え処理を含んでおり、第２のプログラムが実行されると、前記関数呼び出しテーブルから順序変更対象の関数への呼び出し先を待避し、前記関数呼び出しテーブル書き換え処理に基づいて、前記待避した呼び出し先を前記関数呼び出しテーブル内の所定の場所に設定して、関数呼び出しの手順を変更することを特徴とする関数呼び出しの手順を動的に変更する方法。

【請求項２】　実行形式を変更できない第１のプログラムに含まれる関数のうち、順序変更対象の関数が、前記第１のプログラムを使用した第２のプログラムにおいて呼び出される順序が予め判明している場合に、前記第１又は第２のプログラム内の関数の呼び出しを追加することによって、第１のプログラム内の関数を呼び出す順序を前記第２のプログラムの実行中に変更する方法であって、

第２のプログラムにおいて呼び出される関数への呼び出し先が格納される関数呼び出しテーブルを備え、前記追加される関数は再帰的な関数であり、そして、前記第２のプログラムは、前記関数呼び出しテーブルのうち、順序変更対象の関数への呼び出し先が格納されている場所に、予め判明している前記呼び出し順序に基づいて、前記場所に格納された呼び出し先を入れ替えて格納することによって関数呼び出しテーブルを書き換える関数呼び出しテーブル書き換え処理を含んでおり、第２のプログラムが実行されると、前記関数呼び出しテーブルから順序変更対象の関数への呼び出し先を待避し、前記関数呼び出しテーブル書き換え処理に基づいて、前記待避した呼び出し先を前記関数呼び出しテーブル内の所定の場所に設定して、関数呼び出しの手順を変更することを特徴とする関数呼び出しの手順を動的に変更する方法。

【請求項３】　実行形式を変更できない第１のプログラムに含まれる関数のうち、順序変更対象の関数が前記第１のプログラムにおいて呼び出される順序が予め判明している場合に、前記第１のプログラムを使用した第２のプログラムの実行中に、第１のプログラム内の関数を呼出す順序を変更する方法であって、

第２のプログラムにおいて呼び出される関数への呼び出し先が格納される関数呼び出しテーブルを備え、そして、前記第２のプログラムは、前記関数呼び出しテーブルのうち、順序変更対象の関数への呼び出し先が格納されている場所に、予め判明している前記呼び出し順序に基づいて、前記場所に格納された呼び出し先を入れ替えて格納することによって関数呼び出しテーブルを書き換える関数呼び出しテーブル書き換え処理を含んでおり、第２のプログラムが実行されると、前記関数呼び出しテーブルから順序変更対象の関数への呼び出し先を待避し、前記関数呼び出しテーブル書き換え処理に基づいて、前記待避した呼び出し先を前記関数呼び出しテーブル内の所定の場所に設定して、関数呼び出しの手順を変更することを特徴とする関数呼び出しの手順を動的に変更する方法。

【請求項４】　前記関数呼び出しテーブルから順序変更対象の関数への呼び出し先を待避させる前の関数呼び出しテーブルを別途保存しておき、必要に際して、前記待避させる前のテーブルを用いて、待避前の呼び出し先に復帰させることを特徴とする請求項１～請求項３の何れかに記載の関数呼び出しの手順を動的に変更する方法。

【請求項５】　前記関数への呼び出し先を、関数の実行部がそれぞれ格納されたメモリ領域の先頭アドレスとすることを特徴とする請求項１～請求項４の何れかに記載の関数呼び出しの手順を動的に変更する方法。

【請求項６】　実行形式を変更できない第１のプログラムに含まれる関数のうち、順序変更対象の関数が前記第１のプログラムにおいて呼び出される順序が予め判明している場合に、前記第１のプログラムを使用した第２のプログラムの実行中に、第１のプログラム内の関数を呼び出す順序を変更する方法であって、

前記第２のプログラムは、第２のプログラムの実行部が格納されるメモリ領域のうち、順序変更対象の関数の実行部が格納されている場所に、予め判明している前記呼び出し順序に基づいて前記順序変更対象の関数の実行部を入れ替えて格納することによって前記第２のプログラムの実行部を書き換える実行部書き換え処理を含んでおり、前記第２のプログラムが実行されると、実行部書き換え処理に基づいて、前記順序変更対象の関数の実行部を前記第２のプログラムの実行部の所定の場所に設定すると共に、変更前の第２のプログラムの実行部を別途保存しておき、必要に際して前記変更前の第２のプログラムの実行部に復帰させることを特徴とする関数呼び出しの手順を動的に変更する方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明はコンピュータプログ

ラムにおける、関数呼び出しの手順を変更する方法、特にソースコードが開示されていない等、実行形式を変更できないプログラム内の関数呼び出しの手順を動的に変更する方法に関するものである。

【０００２】

【従来の技術】従来、関数呼び出しの手順を変更するには、ソースプログラムを変更して再度実行形式を作成し直すといった静的に変更する方法が行われていた。このため、関数呼び出し手順の変更は、ソースコードが開示され、実行形式を作成し直すことが可能なプログラム（すなわち、自身が作成したプログラムであって、以下、自身のプログラムという）でのみ可能であった。

【０００３】

【発明が解決しようとする課題】一方、近年のプログラム開発では、開発作業を簡略化する等の理由から、別プロダクト（ソースコードを開示しない、若しくはソースコードを開示していてもそれを変更後、実行形式を作成できない関数を提供している部門）が提供するライブラリ（プログラムで使われる関数やデータの集まり）をリンク（結合）することによって実行形式を作成する方法が一般的になってきている。このように、別プロダクトのライブラリ（すなわち、実行形式を変更できないプログラムであって、以下、別プロダクトのプログラムという）を使用して開発された自身のプログラムに、例えばこの別プロダクトのプログラム内の関数（以下、別プロダクトの関数という）からの戻り値が予想外の値であったり、その別プロダクトの関数の動作時間がかかりすぎている等の問題点があったときには、まず、その別プロダクトの関数のバグなのか、その関数が呼び出している別の関数のバグなのかの切り分けを行う必要がある。そこで、この切り分けを行う方法として、別プロダクトの関数を呼び出す手順を所望の手順に変更する方法が考えられている。しかしながら、従来技術では、上記したように自身のプログラム内の関数を呼び出す手順の変更は容易では無いにしろソースプログラムを静的に変更することにより可能であるが、ここで望まれているように別プロダクトの関数を呼び出す手順を所望の手順に変更することは不可能であるという問題点があった。

【０００４】このようなことから、別プロダクトの関数を呼び出す手順を所望の手順に変更することが可能な関数呼び出しの手順を動的に変更する方法の開発が望まれていた。

【０００５】

【課題を解決するための手段】本発明に係る関数呼び出しの手順を動的に変更する方法は、実行形式を変更できない第１のプログラムに含まれる関数のうち、順序変更対象の関数が第１のプログラムにおいて呼び出される順序が予め判明している場合に、第１又は第１のプログラムを使用した第２のプログラム内の関数の呼び出しを追加することによって、第１のプログラム内の関数を呼び出す順序を第２のプログラムの実行中に変更する方法であって、第２のプログラムにおいて呼び出される関数への呼び出し先が格納される関数呼び出しテーブルを備え、追加される関数は再帰的な関数であり、そして、第２のプログラムは、関数呼び出しテーブルのうち、順序変更対象の関数への呼び出し先が格納されている場所に、予め判明している呼び出し順序に基づいて、場所に格納された呼び出し先を入れ替えて格納することによって関数呼び出しテーブルを書き換える関数呼び出しテーブル書き換え処理を含んでおり、第２のプログラムが実行されると、関数呼び出しテーブルから順序変更対象の関数への呼び出し先を待避し、関数呼び出しテーブル書き換え処理に基づいて、待避した呼び出し先を関数呼び出しテーブル内の所定の場所に設定して、関数呼び出しの手順を変更するものである。なお、第１のプログラムとは、後述の実施の形態の別プロダクトのプログラムに相当し、第２のプログラムとは、同様に後述の実施の形態の自身のプログラムに相当する。

【０００６】本発明においては、第２のプログラムの実行中に、関数呼び出しテーブル書き換え処理に基づいて、関数呼び出しテーブルに格納されている先頭アドレスのうち、順序変更の対象である関数の実質の実行部の先頭アドレスを、関数呼び出しテーブル内の所定の場所に設定して、関数呼び出しの手順を動的に変更する。

【０００７】

【発明の実施の形態】図１は本発明の一実施の形態の処理の流れを示すフローチャート、図２は本発明の一実施の形態に係る関数呼び出しの手順を動的に変更する方法が適用されたシステムの構成を示すブロック図である。このシステムにおいて、１は自身のプログラム及び別プロダクトのプログラムを記憶する補助記憶装置、２はＲＡＭ、３はＲＯＭ、４はキーボード等の入力装置、５はＣＰＵであり、入力装置４によりプログラムの実行が指示されると、ＣＰＵ５はプログラムを補助記憶装置１から読み出して、ＲＯＭ３に記憶されたローディングプログラムによりＲＡＭ２に記憶し、プログラムを実行する。

【０００８】次に、関数呼び出し順序変更の種類について説明する。この種類を以下の（ａ）～（ｄ）に示す。なお、以下の説明にある関数Ｘ、関数Ｙ、関数Ｚは、自身のプログラム内の関数、別プロダクトの関数のどちらでも良く、それぞれ任意である。

（ａ）現在の関数呼び出し順序に、新しい関数呼び出しを追加

例えば関数Ｘ→関数Ｚを関数Ｘ→関数Ｙ→関数Ｚの順に変更する場合

（ｂ）関数呼び出し順序変更

例えば関数Ｘ→関数Ｙ→関数Ｚを関数Ｘ→関数Ｚ→関数Ｙの順に変更する場合

（ｃ）関数呼び出し順序の入れ替え

例えば関数Ｘ→関数Ｙを関数Ｘ→関数Ｚの順に変更する場合

（ｄ）関数呼び出し順序の削除

例えば関数Ｘ→関数Ｙ→関数Ｚを関数Ｘ→関数Ｚの順に変更する場合

【０００９】以下、具体的一例を挙げながら本実施の形態の動作を図１のフローチャートを参照しながら説明する。

第１例

ここでは、上記（ａ）に対応する例として、別プロダクトＡの関数Ｘが別プロダクトＢ（別プロダクトＡであっても良い）の関数Ｙを呼び出していることが判明している場合において、この別プロダクトＡ及び別プロダクトＢのプログラムを使用した自身のプログラム実行中、本来ならＸ関数→Ｙ関数の順に行われる呼び出しを、Ｙ関数が実際に実行される直前に、自身のプログラム内の関数であるＢ関数を呼び出し、Ｂ関数の呼び出しの後、Ｙ関数を実行する場合、すなわち、Ｘ関数→Ｂ関数→Ｙ関数の順に変更して呼び出す場合を例に説明する。なお、このように呼び出す順序を変更するための処理（関数呼び出しテーブル書き換え処理）は、予め判明している呼び出し順序に基づいて作成されたものであって、もちろん自身のプログラム内に記述されており、順序変更対象の関数の呼び出しが行われる前に実行される。また、この（ａ）の場合のように、現在の関数呼び出しに新しい関数を追加することにより呼び出し順序を変更する場合には、追加される関数（ここではＢ関数）は再帰的な関数（自分自身を呼び出す関数）でなければならない。そして、自身のプログラムの実行が入力装置４にて指示されると、ＣＰＵ５は自身のプログラムを補助記憶装置１から読み出して、ＲＯＭ３に記憶されたローディングプログラムによりＲＡＭ２に記憶する。

【００１０】図３はこの第１例の自身のプログラムがＲＡＭ上に確保されたメモリ領域の概略を示す図である。図３に示すように、メモリ（ＲＡＭ２）上に格納された自身のプログラムは、プログラム実行部①、プログラム実行部②、関数呼び出しテーブル、別プロダクトＡのプログラムの実行部及び別プロダクトＢのプログラムの実行部の構成になっており、メモリ上の保護領域に確保され、ここで説明するように、自身のプログラムが別プロダクトのプログラムを使用している場合には、別プロダクトのプログラムの実行部は、自身のプログラムの実行部と同一の保護領域に結合され、自身のプログラムの一部として動作する。

【００１１】プログラム実行部①は、実際に処理の流れを記述した部分で、ここでは、関数（自身のプログラムの関数／別プロダクトＡ及びＢの関数含む）呼び出しの順番が記述される。すなわち、呼び出し先の関数に対応する後述の関数呼び出しテーブルのアドレスが呼び出し順に格納される。このため、関数呼び出しテーブルの書き換え処理は、順序変更対象の関数Ｘの呼び出しが行われる前に格納される。プログラム実行部②は、実際に処理の流れを記述した部分で、ここでは、自身が作成した関数群それぞれの実行部が格納される。関数呼び出しテーブルは、呼び出し先を格納した部分で、ここでは、呼び出し先の全ての関数の実質の実行部が格納されたメモリ領域の先頭アドレスが格納される。別プロダクトのプログラムの実行部には、別プロダクトの関数の実質の実行部が格納される。

【００１２】そして、図３に示すようにマッピング後、ＣＰＵ５はプログラムの実行を開始する。まず、プログラム実行部①の関数呼び出しテーブル書き換え処理に基づいて動作を行う。すなわち、順序変更の対象である一方のＹ関数の実質の実行部の先頭アドレスが格納されている関数呼び出しテーブルのアドレス（ここでは、テーブルアドレス）を求め、求めたアドレスの場所に格納されているＹ関数の実質の実行部の先頭アドレスを取得して（図１のＳ１参照）、該先頭アドレスを関数呼び出しテーブルから待避する（図１のＳ２参照）。そして、同様に、もう一方のＢ関数の実質の実行部の先頭アドレスが格納されている関数呼び出しテーブルのアドレス（ここでは、テーブルアドレス）を求め、求めたアドレスの場所に格納されているＢ関数の実質の実行部の先頭アドレスを取得して（図１のＳ３参照）、該先頭アドレスを関数呼び出しテーブルから待避する（図１のＳ４参照）。そして、関数呼び出しテーブルの書き換えを行う。以下、この書き換え動作について説明する。

【００１３】図４は書き換え後のメモリ領域の概略を示した図である。図４に示すように、テーブルアドレス番地にＢ関数の実質の実行部の先頭アドレスを設定し、テーブルアドレス番地にＹ関数の実質の実行部の先頭アドレスを設定し（図１のＳ５参照）、これにより関数呼び出しテーブルの書き換えを行う。

【００１４】以上の関数呼び出しテーブル書き換え処理に基づく動作が終了すると、プログラム実行部①に格納されたテーブル形式関数呼び出しテーブルアドレスにより、関数呼び出しテーブルアドレス番地に格納されたＸ関数の実質の実行部の先頭アドレスが特定され、該特定されたアドレスにあるＸ関数が実行され、Ｘ関数実行部に格納されたテーブルアドレスにより、関数呼び出しテーブルアドレス番地に格納されたＢ関数の実質の実行部の先頭アドレスが特定され、該特定されたアドレスにあるＢ関数が実行される。そして、Ｂ関数実行部に格納されたテーブルアドレスにより、関数呼び出しテーブルアドレス番地に格納されたＹ関数の実質の実行部の先頭アドレスが特定され、該特定されたアドレスにあるＹ関数が実行される。つまり、Ｘ関数→Ｂ関数→Ｙ関数の順に実行されることとなる。

【００１５】図５は第１例を説明するためのメモリマップを示す図であって、図３及び図４の具体例を示してい

る。図５において、（１）は図３に対応し、（２）は図４の関数呼び出しテーブルに対応している。以下、ＲＡＭ２へのローディング後の本実施の形態の動作を図５の具体例に則して再度簡単に説明する。まず、プログラム実行部①の関数呼び出しテーブル書き換え処理に基づいて動作を行う。すなわち、Ｙ関数の実行部の先頭アドレスが格納されている関数呼び出しテーブルのアドレス３００８を求めて３００８番地に格納されている先頭アドレス５２００を待避し、Ｂ関数の実行部の先頭アドレスが格納されている関数呼び出しテーブルのアドレス３０００を求めて３０００番地に格納されている先頭アドレス２０００を待避する。そして、３００８番地にＢ関数の実行部の先頭アドレス２０００を設定し、３０００番地にＹ関数の実行部の先頭アドレス５２００を設定し、これにより関数呼び出しテーブルの書き換えを行う。

【００１６】以上の関数呼び出しテーブル書き換え処理に基づく動作が終了すると、プログラム実行部①内に格納されたテーブルアドレス３００４により、関数呼び出しテーブルの３００４番地に格納されたデータ５１００が特定され、５１００番地にあるＸ関数が実行され、Ｘ関数実行部に格納されたテーブルアドレス３００８により、関数呼び出しテーブルの３００８番地に格納されたデータ２０００が特定され、２０００番地にあるＢ関数が実行される。そして、Ｂ関数実行部に格納されたテーブルアドレス３０００により、関数呼び出しテーブルの３０００番地に格納されたデータ５２００が特定され、５２００番地にあるＹ関数が実行される。つまり、Ｘ関数→Ｂ関数→Ｙ関数の順に呼び出しが行われることとなる。

【００１７】第２例

上記の（ａ）に対応する方法は、自身のプログラム内の関数自体の関数呼び出しの順序を変更する際にも利用できる。例えば自身のプログラムが別プロダクトの関数（自身のプログラム内の関数でもよい）を呼び出している場合に、その関数呼び出しの前に、別関数（この関数は再帰的な関数でなければならないため、自身のプログラム内の関数であることが自然である）を呼び出して、自身のプログラム内の関数自体を呼び出す手順を変更する場合である。もちろんこの場合には、上記の方法を利用しなくてもソースプログラムを変更することにより可能であるが、自身のプログラムが別プロダクトの同一関数を複数箇所呼び出すようにコーディングされていた場合、修正個所が複数箇所となり、個数が多ければ多いほど変更作業が大変になる。そこで、関数呼び出しテーブル書き換え処理を順序変更対象の関数呼び出しが行われる前に加えることにより呼び出し順序を変更することが可能となる。このため修正の手間を大幅に省くことができる。

【００１８】以下、具体例で説明する。例えば、自身のプログラム内の関数Ｂ、関数Ｃ、関数Ｄが、それぞれ別プロダクトＡの関数Ｘを呼び出す関数である場合に、関数Ｘの呼び出しの前に自身のプログラム内の関数Ｙ（再帰的な関数）の呼び出しを行いたい場合、すなわち、Ｂ関数→Ｙ関数→Ｘ関数、Ｃ関数→Ｙ関数→Ｘ関数、Ｄ関数→Ｙ関数→Ｘ関数の順に呼び出したい場合について簡単に説明する。図６はこの場合のメモリマップを示す図である。（３）は変更前のメモリマップを示しており、（４）は変更後の関数呼び出しテーブルを示している。

【００１９】自身のプログラムが実行されると、関数呼び出しテーブル書き換え処理に基づいて動作が行われ、関数呼び出しテーブルの３０１２番地に格納されているＹ関数の実行部の先頭アドレス２３００と、関数呼び出しテーブルの３０１６番地に格納されているＸ関数の実行部の先頭アドレス５１００とが入れ替えて関数呼び出しテーブルに格納される。そして、プログラム実行部①内に格納されたテーブルアドレス３０００により、Ｂ関数が実行され、そして、Ｂ関数実行部に格納されたテーブルアドレス３０１６によりＹ関数が実行され、Ｙ関数実行部に格納されたテーブルアドレス３０１２によりＸ関数が実行される。つまり、Ｂ関数→Ｙ関数→Ｘ関数の順に呼び出しが行われることとなる。なお、Ｃ関数及びＤ関数の場合も同様であるので、説明を省略する。

【００２０】第３例

以下、上記（ｂ）に対応する例として、例えば自身のプログラムは別プロダクトＡのＸ関数を呼び出しており、該Ｘ関数は別プロダクトＢのＹ関数及びＺ関数を有し、この順に呼び出す関数であることが判明している場合において、自身のプログラム内でこの別プロダクトＡの関数Ｘを呼び出す際に、本来ならＸ関数→Ｙ関数→Ｚ関数の順に行われる呼び出しを、Ｘ関数→Ｚ関数→Ｙ関数の順に変更して呼び出す場合を例に説明する。なお、このように呼び出す順序を変更するための処理（関数呼び出しテーブル書き換え処理）は、第１及び第２例の場合と同様に、予め判明している呼び出し順序に基づいて作成されたものであって、もちろん自身のプログラム内に記述されており、順序変更対象の関数の呼び出しが行われる前に実行される。

【００２１】図７はこの第３例を説明するためのメモリマップを示す図であって、（５）は変更前のメモリマップを示しており、（６）は変更後の関数呼び出しテーブルを示している。そして、図７に示すようにマッピング後、ＣＰＵ５はプログラムの実行を開始する。まず、プログラム実行部①の関数呼び出しテーブル書き換え処理に基づいて動作を行う。すなわち、順序変更の対象である一方のＹ関数の実質の実行部の先頭アドレスが格納されている関数呼び出しテーブルのアドレス３００４を求め、３００４番地に格納されているＹ関数の実質の実行部の先頭アドレス６１００を取得して、該先頭アドレスを関数呼び出しテーブルから待避する。そして、同様にもう一方のＺ関数の実質の実行部の先頭アドレスが保存

されている関数呼び出しテーブルのアドレス３００８を求め、３００８番地に格納されているＺ関数の実質の実行部の先頭アドレス６２００を取得して、該先頭アドレスを関数呼び出しテーブルから待避する。そして、３００４番地にＺ関数の実質の実行部の先頭アドレス６２００を設定し、３００８番地にＹ関数の実質の実行部の先頭アドレス６１００を設定する。

【００２２】以上の関数呼び出しテーブル書き換え処理に基づく動作が終了すると、プログラム実行部①の１０００番地に格納されたテーブルアドレス３０００により、関数呼び出しテーブルの３０００番地に格納されたデータ５１００が特定され、５１００番地にあるＸ関数が実行される。すなわち、Ｘ関数実行部の５１０４番地に格納されたテーブルアドレス３００４により、６２００番地にあるＺ関数が実行され、そして、Ｘ関数実行部の５１０８番地に格納されたテーブルアドレス３００８により、６１００番地にあるＹ関数が実行される。つまり、Ｘ関数→Ｚ関数→Ｙ関数の順に呼び出しが行われることとなる。

【００２３】第４例

以下、上記（ｃ）を関数Ｘ→関数Ｙを関数Ｘ→関数Ｚの順に変更する場合、（ｄ）を関数Ｘ→関数Ｙ→関数Ｚを関数Ｘ→関数Ｚの順に変更する場合を例に説明する。なお、これら（ｃ）、（ｄ）の場合は、上記に説明した場合と同様であるので、ここでは簡単に説明する。自身のプログラムは、関数呼び出しテーブルに格納されている、順序変更の対象である関数Ｙと関数Ｚの実質の実行部の先頭アドレスを入れ替えて前記関数呼び出しテーブルに設定する関数呼び出しテーブル書き換え処理を含んでおり、そして、当該自身のプログラムが実行されると、前記書き換え処理に基づいて関数呼び出しテーブルが書き換えられて、関数呼び出しの手順が変更される。

【００２４】本実施の形態によれば、自身のプログラムの実行中に、順序変更対象の関数実行部がそれぞれ格納されたメモリ領域の先頭アドレスが格納された関数呼び出しテーブルのアドレスを求め、該アドレス内の先頭アドレスを関数呼び出しテーブルから待避し、待避した先頭アドレスを、関数呼び出しテーブル書き換え処理に基づいて関数呼び出しテーブルの所定の場所に設定することによって、関数呼び出しテーブルを動的に書き換えることができる。その結果、関数呼び出しテーブルを使用して関数呼び出しが行われる際、別プロダクト内の関数の順序が変更して呼び出されることとなり、従来の関数呼び出しの手順を変更する方法では実現し得なかった、別プロダクト内の関数を呼び出す手順を動的に変更することが可能となる。

【００２５】また、本実施の形態の方法を自身のプログラムに当てはめて実施すれば、自身のプログラム内の関数呼び出しの順序変更自体も可能となり、また、上記第２例のように、自身のプログラムがある関数を複数箇所呼び出すようにコーディングされていた場合に、この関数の前に別の関数を呼び出すように呼び出し手順を変更する際には、自身のプログラムの複数箇所を修正することなく複数箇所の関数呼び出しの手順を変更でき、修正の手間を大幅に省くことができる。

【００２６】なお、関数呼び出しテーブルを書き換えて関数呼び出しの手順を変更した後に、何らかの条件の基で変更前の元の関数呼び出しを行う必要がある場合には、変更前の関数呼び出しテーブルを自身のプログラムの実行部の同一保護領域に別途保存しておき、必要となったタイミングで変更前の関数呼び出しテーブルを使用して元の関数呼び出しに使用する。

【００２７】また、本実施の形態では、呼び出し先を格納した関数呼び出しテーブルを書き換える方法を説明したが、別プロダクトのプログラムの実行部が格納されたメモリ領域のうち、順序変更対象の関数の実行部が格納されている場所に、予め判明している呼び出し順序に基づいて前記順序変更対象の関数の実行部を入れ替えて格納することによって別プロダクトのプログラムの実質の実行部を書き換え、関数呼び出しの手順を変更するようにしても良い。この場合、別プロダクトの実質の実行部自体が書き変わってしまうことになるので、関数呼び出しの手順変更後に何らかの条件の基で変更前の元の関数呼び出しを行う必要がある場合には、この別プロダクトの実質の実行部の複製を自身のプログラムの実行部の同一保護領域に作成しておき、必要となったタイミングで呼び出すようにする。

【００２８】また、関数呼び出しテーブルを書き換えること、又は上記したように別プロダクトの実質の実行部を書き換えることによって、別プロダクトのプログラムの関数部分の実行を抑止したり、関数の機能を変更することが可能になる。

【００２９】

【発明の効果】以上に説明したように本発明によれば、第２のプログラムの実行中に、順序変更対象の関数への呼び出し先を、関数呼び出しテーブル書き換え処理に基づいて関数呼び出しテーブルの所定の場所に設定することによって、関数呼び出しテーブルを動的に書き換えることができる。その結果、関数呼び出しテーブルを使用して関数呼び出しが行われる際、第１のプログラム内の関数の順序が変更して呼び出されることとなり、従来の関数呼び出しの手順を変更する方法では実現し得なかった、第１のプログラム内の関数を呼び出す手順を動的に変更することが可能となる。

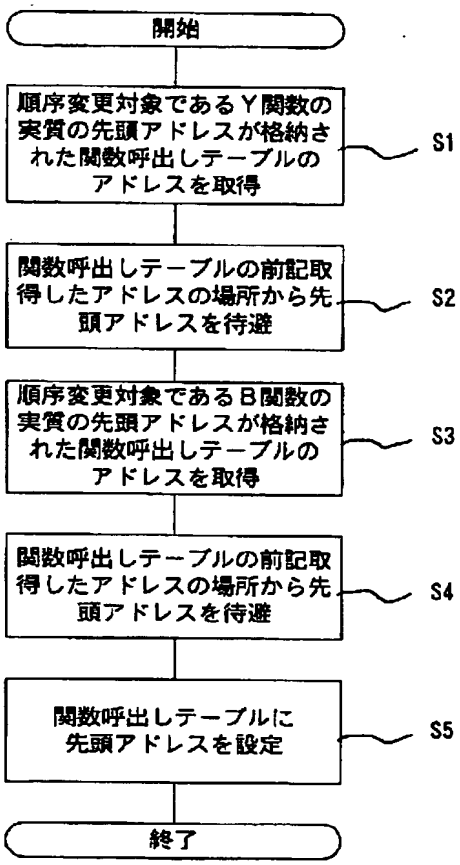
【図面の簡単な説明】

【図１】本発明の一実施の形態の処理の流れを示すフローチャートである。

【図２】本発明の一実施の形態に係る関数呼び出しの手順を動的に変更する方法が適用されたシステムの構成を示す図である。

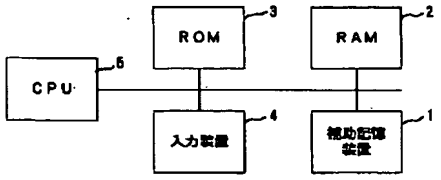
【図３】第１例の自身のプログラムがＲＡＭ上に確保されたメモリ領域の概略を示す図である。
【図４】図３を書き換え後のメモリ領域の概略を示した図である。
【図５】第１例を説明するためのメモリマップを示す図である。
【図６】第２例を説明するためのメモリマップを示す図である。
【図７】第３例を説明するためのメモリマップを示す図である。
【符号の説明】
1　補助記憶装置
2　ＲＡＭ
3　ＲＯＭ
4　入力装置
5　ＣＰＵ

【図１】



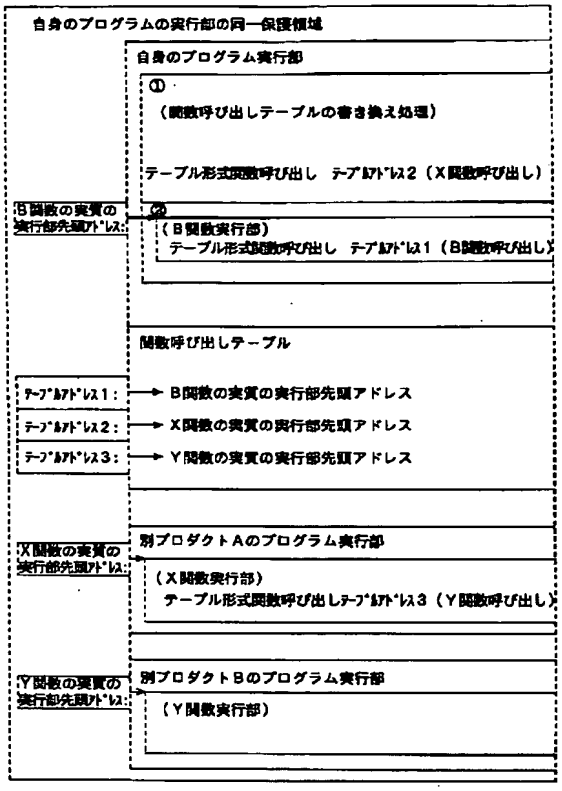
本発明の一実施の形態の処理の流れを示すフローチャート

【図２】



本発明の一実施の形態に係る方法が適用されたシステムの構成を示す図

【図３】



第１例の自身のプログラムがＲＡＭ上に確保されたメモリ領域の概略を示す図

【図４】

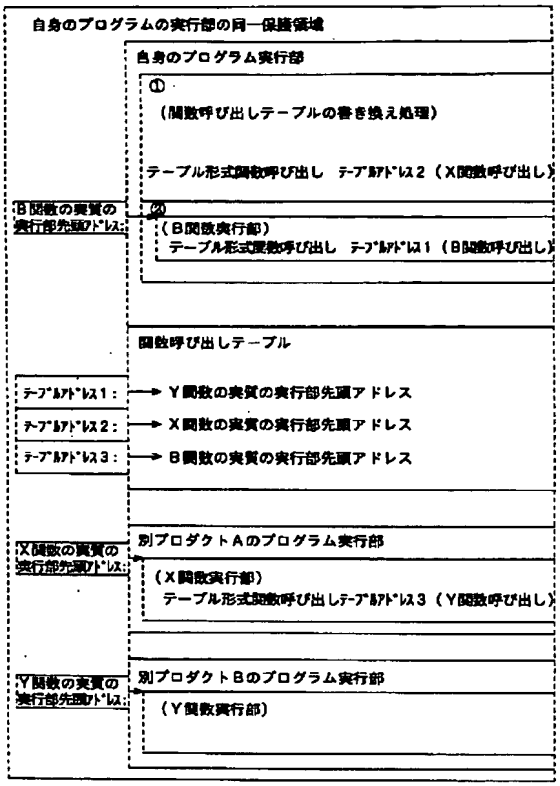
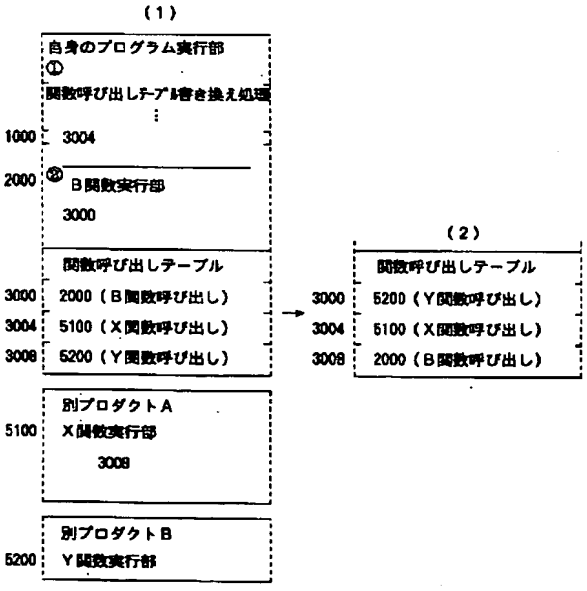


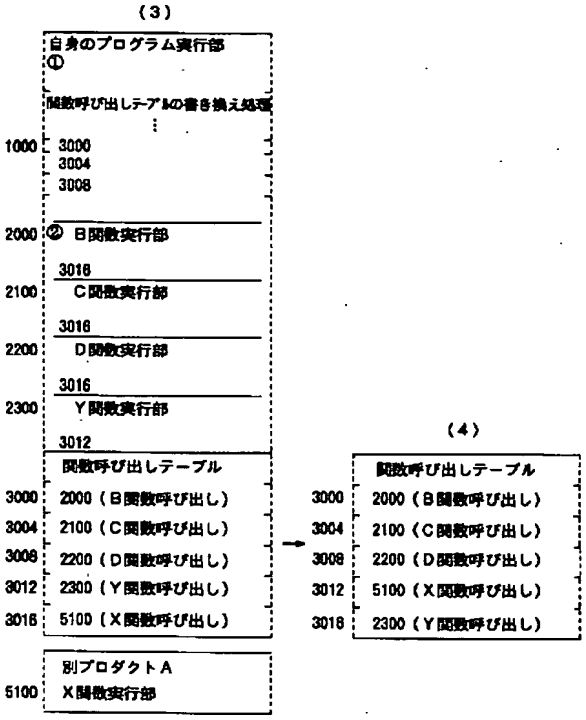
図３を書き換え後のメモリ領域の概略を示す図

【図５】



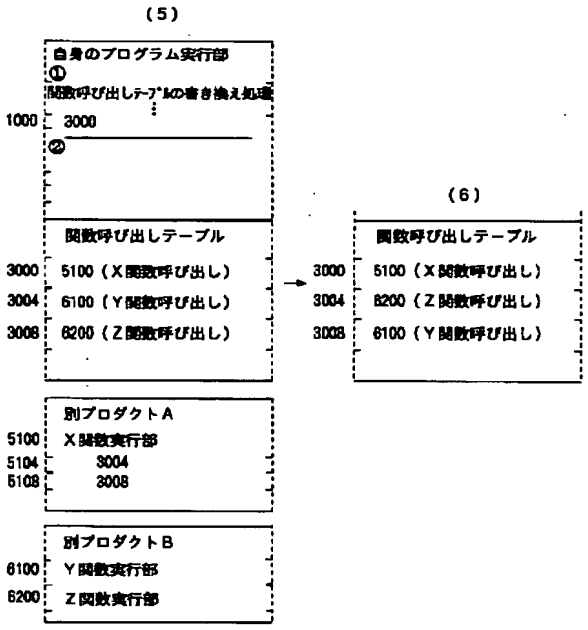
第１例を説明するためのメモリマップを示す図

【図６】



第２例を説明するためのメモリマップを示す図

【図７】



第３例を説明するためのメモリマップを示す図

フロントページの続き

(72)発明者　白根　恭子
岡山県岡山市桑田町18番28号　株式会社沖ソフトウェア岡山内

(72)発明者　重広　正夫
東京都港区虎ノ門１丁目７番12号　沖電気工業株式会社内

Ｆターム(参考)　5B076 EA10 EA17